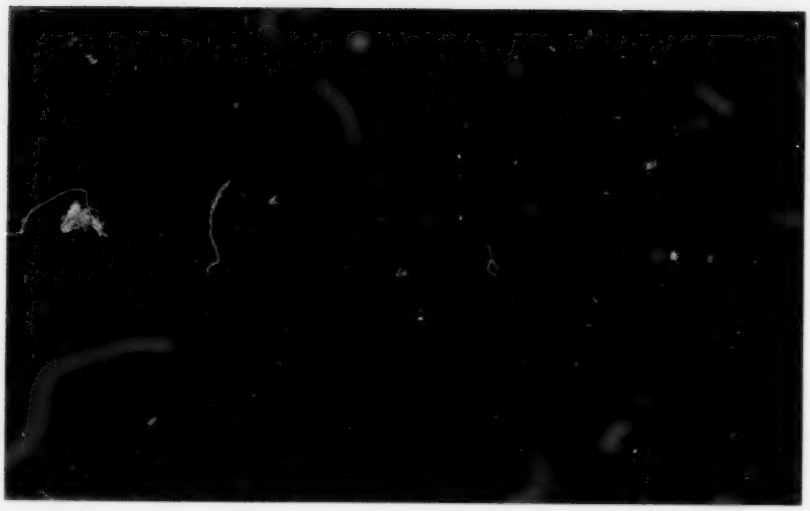
CENTRE DE RECHERCHE SUR LES TRANSPORTS
(C.R.T.)
CENTRE FOR RESEARCH ON TRANSPORTATION

UNIVERSITÉ DE MONTRÉAL
ÉCOLE DES HAUTES ÉTUDES COMMERCIALES
ÉCOLE POLYTECHNIQUE



CENTRE DE RECHERCHE SUR LES TRANSPORTS
UNIVERSITÉ DE MONTRÉAL
C.P. 6128, SUCCURSALE A,
MONTRÉAL
CANADA H3C 3J7

TÉLÉPHONE: (514) 343-7575
TÉLEX: (21) 05-561325
EMAIL: crt@crt.umontreal.ca
TÉLÉCOPIEUR (FAX): (514) 343-7121

91-04083/

CRT-766

A Projective Method for Bipartite Networks and Application to the Matrix Estimation and Transportation Problems

by
Omar Drissi-Kaïtouni

This research was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada.

Centre de recherche sur les transports — Publication #766

April 1991

Dépôt légal – 2ème trimestre 1991

Abstract

This paper deals with bipartite network problems where some objective function is to be optimized under flow conservation constraints at nodes, and, eventually, bounds on the link flows. Such problems are also interpreted as matrix optimization problems. In this paper, we first construct a linear complexity projector onto the feasible space defined by the flow conservation constraints. Then, we develop a variant of the scaled conjugate gradient method, where the conjugate directions are all forced to lie in the feasible space; projections that are required at each iteration of this projected scaled conjugate gradient algorithm, are performed by the efficient projector developed earlier. Numerical results support superiority of this approach against classical conjugate gradient and LDL decomposition techniques for solving the problem.
Key words: Bipartite, Network, Projection, Conjugate Gradient.

Résumé

Cet article traite de problèmes de réseaux bipartites où une fonction objectif doit être optimisée sous contraintes de conservation de flot, et, éventuellement, bornes sur les flots. Ces problèmes s'interprètent aussi comme des problèmes d'optimisation de matrice. Nous construisons, dans cet article, un projecteur, de complexité linéaire, sur l'espace réalisable défini par les contraintes de conservation de flot. Ensuite, nous développons une variante de la méthode du gradient conjugué où les directions conjuguées sont forcées d'être dans l'espace réalisable; les projections requises sont effectuées par le projecteur efficace développé précédemment. Les résultats numériques supportent la supériorité de cette approche par rapport aux méthodes classiques de décomposition LDL et de gradient conjugué.
Mots clés: Bipartite, réseau, projection, gradient conjugué.

1. INTRODUCTION

In this paper, we consider a bipartite network $G = (N, L)$, where N is the set of nodes and L the set of links of G. The set of nodes is partitionned into two disjoint subsets O and D, where O is the set of origin nodes of cardinality $|O| = n$, and D is the set of destination nodes of cardinality $|D| = m$. The set of links L consists of all the pairs $a = (i, j)$, $i = 1, .., n$, $j = n + 1, .., n + m$; its cardinality is mn.
The node-link incidence matrix $\bar{A}$ for the bipartite network G is given by:

$$\bar{A} = \begin{array}{c|cccccccccccc} & 1 & . & 1 & 2 & . & 2 & . & n & . & n \\ & n+1 & . & n+m & n+1 & . & n+m & . & n+1 & . & n+m \\ \hline 1 & -1 & \dots & -1 & & & & & & & \\ 2 & & & & -1 & \dots & -1 & & & & \\ \dots & & & & & & & \dots & \dots & \dots & \\ n & & & & & & & & -1 & \dots & -1 \\ n+1 & 1 & & & 1 & & & \dots & & 1 & \\ \dots & & \dots & & & \dots & & & \dots & & \\ n+m & & & 1 & & & 1 & & \dots & & 1 \end{array}$$

As for any network, each column (i, j), $i = 1, .., n$, $j = n+1, .., n+m$, of $\bar{A}$ has exactly two nonzero entries that are -1 at position i and $+1$ at position j. Thus, summing all the rows of $\bar{A}$ leads to a zero row. Therefore, one row is a linear combination of all the others. The *truncated* node-link incidence matrix A is matrix $\bar{A}$ from which one row is deleted; matrix A is then of full rank $n + m - 1$. In the remainder of this paper, we refer to A as being $\bar{A}$ from which the *last* row $n + m$ is removed.

Problems considered in this paper, are of the form (1), (2), (3) and (4), that is written in short in (5), (6) and (7). This problem, and some of its variants, is the mathematical formulation of many applications in practice such the convex transportation problem and its spatial price equilibrium variants ([4], [5], [13], [16]), origin destination demand matrices in network equilibrium and input output matrices in economics ([2]), point to point estimation in telecommunication ([9]), water recycling ([15]), and so on.

$$\min \quad f(x) = \sum_{i=1}^{n}\sum_{j=1}^{m} f_{ij}(x) \tag{1}$$

$$s.t. \quad -\sum_{j=1}^{m} x_{ij} = -S_i \qquad i = 1,..,n \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = D_j \qquad j = 1,..,m-1 \tag{3}$$

$$x_{ij} \geq 0 \qquad i = 1,..,n, \quad j = 1,..,m \tag{4}$$

$$\min \quad f(x) \tag{5}$$

$$s.t. \quad Ax = b \tag{6}$$

$$x \geq 0 \tag{7}$$

Dual based exact and inexact Newton solution approaches ([8], [9]) have been proposed in literature to solve this problem. Other methods based on Lagrangean and relaxation techniques, was proposed for the quadratic case ([2], [10], [12]). Since the above convex transportaion problem is a special case of general network flow problems, the algorithms (Frank-Wolfe, reduced gradient, approximative Newton,...) proposed for the latter problem ([6], [8], ...) are suitable for solving the former; these methods take advantage of the network structure of the problem, but they do not exploit the inherent network symmetries in a complete bipartite network.

In section 2, we show that network symmetries of a bipartite network, may be exploited efficiently in vector projections. These results are used in section 3 to develop interesting projection based algorithms for the convex transportation problem. Numerical results are presented in section 4, and the paper is concluded in section 5.

2. AN ANALYTIC PROJECTION METHOD FOR BIPARTITE NETWORKS

This section deals with the orthogonal projection $p = Pu$ of a vector $u \in R^{nm}$ onto the null space of the truncated incidence matrix A of a bipartite network. The projection is obtained by solving the quadratic problem (8)-(9):

$$\min \quad \frac{1}{2}(p-u)^2 \tag{8}$$

$$s.t. \quad Ap = 0 \tag{9}$$

Kuhn-Tucker optimality conditions for problem (8)-(9) are:

$$p - u + A^T\lambda = 0 \tag{10}$$

$$Ap = 0 \tag{11}$$

where $\lambda \in R^{n+m-1}$. The solution of this system is obtained as follows: from (10):

$$p = u - A^T\lambda \tag{12}$$

and, multipying (12) by A and using (11), we obtain (13), and equivalently the system of linear equations (SLE) (14) that must be solved in order to find λ:

$$0 = Ap = A(u - A^T\lambda) = Au - AA^T\lambda \tag{13}$$

$$AA^T\lambda = v \tag{14}$$

where, by definition,

$$v = Au \tag{15}$$

Once λ is computed by solving (14), p is computed by using (12). Note that this leads to the classical projector $P = (I - A^T(AA^T)^{-1}A)$, since $p = u - A^T\lambda = u - A^T(AA^T)^{-1}(Au) = Pu$. In the context of bipartite networks, (12) leads to a very efficient way for computing p, that is:

$$p_a = u_a + \lambda_i - \lambda_j \quad \forall a = (i,j), i = 1,..,n, j = n+1,..,n+m \tag{16}$$

where λ_{n+m} is set to zero.

Therefore, when the vector of dual variables λ is known, the number of operations required for the computation of p is in the order of the number of links of the bipartite network. In the remainder of this section, we concentrate on the solution of the SLE (14).

The right term of (14), that is $v = Au$ is easily computed by:

$$v_l = \begin{cases} -\sum_{j=n+1}^{n+m} u_{a=(l,j)} & \text{if } l = 1,..,n \\ \sum_{i=1}^{n} u_{a=(i,l)} & \text{if } l = n+1,..,n+m-1 \end{cases} \tag{17}$$

The solution λ of (14) is then given by Theorem 1. Note first that AA^T is:

$$AA^{\top} = \begin{array}{|ccc|ccc|} \hline m & & & -1 & . & -1 \\ & . & & . & . & . \\ & & m & -1 & . & -1 \\ \hline -1 & . & -1 & n & & \\ . & . & . & & . & \\ -1 & . & -1 & & & n \\ \hline \end{array}$$

THEOREM 1: *The solution* λ *of system* (14): $AA^{\top}\lambda = v$, *where* A *is the truncated node-link incidence matrix of a bipartite network* $G=(N,L)$ *defined above is:*

$$\lambda_j = \begin{cases} \frac{1}{n}(\bar{v} - \frac{1}{m}\bar{v} + \frac{n}{m}v_j) & \text{if } j = 1,..,n \\ \\ \frac{1}{n}(\bar{v} + v_j) & \text{if } j = n+1,..,n+m-1 \end{cases} \tag{18}$$

where,

$$\bar{v} = \sum_{i=1}^{n} v_i \tag{19}$$

and,

$$\bar{\bar{v}} = \sum_{i=1}^{n+m-1} v_i = \bar{v} + \sum_{j=n+1}^{n+m-1} v_j \tag{20}$$

PROOF: In the appendix.

In order to count the number of operations required to compute λ, we rewrite equations (11), (12) and (13), in an algorithmic form:

Computation of $\lambda = (AA^{\top})^{-1}v$:

STEP 1: Compute $\bar{v} = \sum_{i=1}^{n} v_i$ (n additions)

STEP 2: Compute $\bar{\bar{v}} = \bar{v} + \sum_{j=n+1}^{n+m-1} v_j$ ($m-1$ additions)

STEP 3: Compute $\alpha = \bar{\bar{v}}/n$ (1 division)

STEP 4: Compute $\beta = (\bar{\bar{v}} - \bar{v}/n)/n$ (1 addition, 2 divisions)

STEP 5: For each $j = 1,..,n$ do: (n iterations)

$\lambda_j = \beta + v_j/m$ (1 addition, 1 division)

STEP 6: For each $j = n+1,..,n+m-1$ do: ($m-1$ iteration)

$\lambda_j = \alpha + v_j/n$ (1 addition, 1 division)

The total number of operations is:

Additions (substractions) : $n+m-1+1+n+m-1 = 2(n+m-1)+1$

Multiplications (divisions) : $1+2+n+m-1 = n+m+2$

Therefore, the computation of λ is done in $O(n+m)$ elementary operations, which is much faster than all what has been proposed in literature; the best proposed method, in our knowledge, is the one described in [9], that is in $O(n+m+\min(m^3,n^3))$; unless $n << m$ or $m << n$, our method is clearly much more faster than the latter.

3. APPLICATION TO THE TRANSPORTATION AND MATRIX ESTIMATION PROBLEMS

In this section, we consider the general convex formulation of the transportation problem (21) to (24). However, results developed in this paper extend to other bipartite network problems such as the spatial price equilibrium problem ([4], [5], [13], [16]).

$$\min \quad f(x) = \sum_{i=1}^{n}\sum_{j=1}^{m} f_{ij}(x) \tag{21}$$

$$s.t. \quad -\sum_{j=1}^{m} x_{ij} = -S_i \qquad i = 1,..,n \tag{22}$$

$$\sum_{i=1}^{n} x_{ij} = D_j \qquad j = 1,..,m-1 \tag{23}$$

$$x_{ij} \geq 0 \qquad i = 1,..,n, \quad j = 1,..,m \tag{24}$$

In the context of transportation problems, variable x_{ij} refers to the flow on the link that has node i as tail node, and node $n+j$ as head node, and S_i and D_j to supply and demand of flow at nodes i and j. Objective function (21) expresses that link flows are to be shipped at minimum total cost, and constraints (22) and (23) assure that flow conservation is satisfied at the origin and destination nodes. In this formulation, we assume that the cost functions are convex and in C^1.

When (21) to (24) is the mathematical formulation of another problem, interpretation of variables will be different; for instance, in matrix estimation problems, a matrix $x = (x_{ij})$: $n \times m$, must be estimated in order that:

i- sums of rows and columns must be equal to the corresponding S_i and D_j's, and

ii- x must be choosen to be as close as possible to some known matrix $\bar{x} = (\bar{x}_{ij})$ (that may

represent past observed values of x). This will often be expressed by the objective function:

$$\min \quad f(x) \;=\; \sum_{i=1}^{n}\sum_{j=1}^{m} f_{ij}(x) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}(x_{ij} - \bar{x}_{ij})^{\nu} \tag{25}$$

where $\nu \geq 2$, but other functionnal forms may be more suitable to model some specific problems ([14]).

In the remainder of this section, we discuss discuss various solution methods for the above problem by relaxing at a first time nonnegativity constraints (24) in section 3.1, and by reconsidering them in section 3.2.

3.1 UNCONSTRAINED MATRIX ESTIMATION

In many practical situations, such as the point to point traffic estimation problem that arize in telecommunication, most of the variables x_{ij} are positive in the optimal solution of (21) to (24); therefore, in this context, nonnegativity constraints may simply be disregarded. The concern of this subsection will be the solution of problem (21) to (23), that we rewrite in a compact form as (26) s.t. (27) where (27) stands for (22) and (23); note that $b_i = -S_i,\ i = 1,..,n$ and $b_{n+j} = D_j,\ j = 1,..,m-1$.

$$\min \quad f(x) \tag{26}$$

$$s.t. \quad Ax = b \tag{27}$$

The algorithmic scheme that we adopt for solving (26) s.t. (27), is the feasible descent direction approach that is summarized in the following:

- [STEP 0:] Initialization: Find an initial feasible solution x
- [STEP 1:] Feasible descent direction: Find a feasible descent direction p, e.g., such that $p^{\top}g < 0$ and $Ap = 0$, where g is the gradient of the objective function f, evaluated at the current solution x.
- [STEP 2:] Linesearch: Find an optimal step α that minimizes the objective function along p; e.g., such that $f(x + \alpha p)$ is minimum according to α.
- [STEP 3:] Update: Set $x = x + \alpha p$.

- [STEP 4:] Termination test: Stop if some stopping criterium is satisfied, otherwise return to Step 1.

Now, let's describe in detail steps 0 and 1 of this scheme; steps 2, 3 and 4 need no detailed description since no very special adaptation of classical methods is required. In the following, we omit to put an iteration index on x, p,..., since no confusion can occure.

STEP 0: Initialization:

We need to find an initial feasible solution x, e.g, such that $Ax = b$. This may be found either by projecting the null vector onto the space $\{x|Ax = b\}$, or, if some matrix $\bar{x}$ is known and that x is attempted to be close to $\bar{x}$, then $\bar{x}$ may be projected onto the space $\{x|Ax = b\}$. We discuss only of the projection of $\bar{x}$ (which includes the particular case $\bar{x} = 0$) onto the space $\{x|Ax = b\}$. This is obtained by solving the following problem:

$$\min \frac{1}{2}(x - \bar{x})^2 \quad s.t. \quad Ax = b \tag{28}$$

The solution of this problem must satisfy:

$$x - \bar{x} + A^T\lambda = 0 \tag{29}$$

$$Ax = b \tag{30}$$

Multiplying (29) by A and replacing Ax by b, leads to:

$$AA^T\lambda = A\bar{x} - b \tag{31}$$

This system may be solved easily by the results obtained in the previous section.

$$\text{Let} \quad \bar{x}_{i.} = \sum_{j=1}^{m-1} \bar{x}_{ij} \quad \text{and} \quad \bar{x}_{.j} = \sum_{i=1}^{n} \bar{x}_{ij} \tag{32}$$

$$\text{and} \quad \overline{S} = \sum_{i=1}^{n}(S_i - \bar{x}_{i.}) \quad \text{and} \quad \overline{D} = \sum_{j=1}^{m-1}(-D_j + \bar{x}_{.j}) \tag{33}$$

Then, according to (14) and (31), $v \equiv A\bar{x} - b$, and according to (19)-(20) and (32)-(33), $\bar{v} = \overline{S}$, and $\bar{v} = \overline{S} + \overline{D}$. Therefore, (18) is rewritten:

$$\lambda_j = \begin{cases} \frac{1}{n}(\overline{S} + \overline{D} - \frac{1}{m}\overline{S} - \frac{n}{m}S_j) & \text{if } j = 1,..,n \\ \\ \frac{1}{n}(\overline{S} + \overline{D} + D_{j-n}) & \text{if } j = n+1,..,n+m-1 \end{cases} \tag{34}$$

Having in mind the special structure of A, equation (29) is used to compute x:

$$x = \bar{x} - A^{\top}\lambda = (\bar{x}_{ij} + \lambda_i - \lambda_{n+j})_{\substack{i=1,..,n \\ j=1,..,m}} \tag{35}$$

where $\lambda_{n+m} = 0$, and $\lambda_j,\ j = 1,..,n+m-1$ are given by (34).

We must note here that if the objective function (26) is of the form corresponding to a matrix estimation problem:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{m} f_{ij}(x) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}(x_{ij} - \bar{x}_{ij})^{\nu}$$

where ν is taken to be $\nu = 2$, then problem (28) is equivalent to (26)-(27), and, therefore, the above *initial* solution (34)-(35) is also the *optimal* solution, and the solution process stops here. For other functional forms of the objective function, the process goes on with step 1:

STEP 1: Feasible descent direction:

In this subsection, we discuss the adaptation of classical first and second order descent direction methods.

- *Projected Gradient*:

Let g be the gradient of the objective function, evaluated at the current feasible solution x. if we take the feasible descent direction p to be the projection of $-g$ onto the null space of A, then p is computed by equation (16) and the related equations in section 2 where u is replaced by $-g$.

In particular, note that, equations (10) and (11) are:

$$p - (-g) + A^{\top}\lambda = 0 \tag{36}$$

$$Ap = 0 \tag{37}$$

and, if (36) is multiplyed by p and Ap is replaced by 0, then

$$p^{\top}p + p^{\top}g + 0 = 0, \quad \text{so,} \quad p^{\top}g = -p^{\top}p < 0 \ \text{ if } \ p \neq 0 \tag{38}$$

Further, if $p = 0$, we have:

$$g + A^{\top}\lambda = 0 \tag{39}$$

$$Ax = b \tag{40}$$

that are the Kuhn-Tucker optimality conditions of the (convex) problem. Therefore, p is a feasible descent direction unless the current solution is optimal.

This approach is an adaptation of the well known projected gradient method that has a linear rate of convergence ([11]); this is much better than many other proposed algorithms such the Frank-Wolfe method that has a sublinear rate of convergence ([7]). Other approches with a higher rate of convergence (Newton method, with a quadratic rate of convergence, [9]) require however a larger number of operations per iteration:

- *Newton Method*:

The Newton direction is obtained by solving the quadratic approximation of the problem at the current solution, that is:

$$\min \quad g^\top p + \frac{1}{2} p^\top H p \quad s.t. \quad Ap = 0 \tag{41}$$

where g and H are respectively the gradient and the hessian matrix of the objective function, evaluated at the current solution. We assume in the following that H is a diagonal positive definite matrix (which holds under usual strict convexity and separability conditions on the objective function). The optimal solution of problem (41) satisfies the Kuhn-Tucker conditions:

$$g + Hp + A^\top \lambda = 0 \tag{42}$$

$$Ap = 0 \tag{43}$$

where p and λ are obtained by:

$$\lambda = -(AH^{-1}A^\top)^{-1} AH^{-1} g \tag{44}$$

$$p = -H^{-1}(g + A^\top \lambda) \tag{45}$$

Direction p is feasible since it satisfies (43). Further, if $p \neq 0$, then p is a descent direction since:

$$p^\top (g + Hp + A^\top \lambda) = 0 \quad \equiv \quad p^\top g = -p^\top H p < 0 \tag{46}$$

Otherwise, $p = 0$ implies that the current solution is optimal since (42)-(43) would lead to (39)-(40).

Klinckewicz [9] shows that system (44) (of size $(n+m-1)\times(n+m-1)$) reduces to the solution of a dense, nontrivial system of m linear equations (supposedly $m \leq n$), and he suggests to solve this system by an LDL decomposition technique that is $O(m^3)$. His results show that this method is better than the conjugate gradient (CG) method ([11]). As an alternative that we will use in our numerical experiments, is the scaled variant of the conjugate gradient algorithm (SCG) discribed in [1]. The SCG method is simply the CG method applied to a pre-conditioned system $\tilde{M}\tilde{\lambda} = \tilde{b}$ (that is equivalent to the original system $M\lambda = b$), where $\tilde{M} = C^{-1}MC^{-1}$, $M = AHA^T$, $\tilde{\lambda} = C\lambda$ and $\tilde{b} = C^{-1}b = -C^{-1}AH^{-1}g$. Matrix C that is a pre-conditioning matrix, must be positive definite and symmetric. Note however that only $S = (C^2)^{-1}$ needs to be known explicitly in the algorithm ([1]). When $S = I$ is the identity matrix, the SCG method turns to be simply the CG method. Both CG and SCG methods require at most m iterations to solve a system of m linear equations; this is due to the fact that at each iteration, the (quadratic) objective is minimized in the subspace generated by the previous conjugate directions, and, as a consequence, the gradient at this point is orthogonal to that subspace; this allows the expansion of the minimization subspace by one dimension at each iteration. In practice, since the SCG works on a pre-conditioned system, this method converges more fastly to accurate solutions, than the classical CG algorithm. However, we must insist on the fact that the solution λ of (44) must be accurate enough to assure feasibility of the direction p in equation (45). Obtaining a quiet accurate solution λ may require many iterations of the SCG (or CG) algorithm. Therefore, as an alternative approach that we will evaluate also, is a projected scaled conjugate gradient (PSCG) method where we force the (scaled) conjugate directions to lie in the null space of A. The interest of such a method is that the process may be terminated at any iteration (even at the first one) with a feasible descent direction.

- *PSCG Method*:

Let $P = (I - A^T(AA^T)^{-1}A)$ be the orthogonal projection matrix on the null space of A. Then any vector p such that $Ap = 0$ may be rewritten as $p = Pz$: if $p = Pz$, then $Ap = APz = (A - AA^T(AA^T)^{-1}A)z = 0$, and reversely, if $Ap = 0$, then with $z = p$, $Pp = p - A^T(AA^T)^{-1}Ap = p - 0 = p$. Therefore, we use this observation to rewrite the

quadratic problem (41) in the following equivalent form:

$$\min \; g^T Pz + \frac{1}{2}(Pz)^T HPz \;=\; z^T(Pg) + \frac{1}{2}z^T(PHP)z \;=\; \bar{g}^T z + \frac{1}{2}z^T \bar{H} z \qquad (47)$$

It is easy to check the equivalence between (41) and (47): if p satisfies (42)-(43), then $0 = g + Hp + A^T\lambda = g + HPp + A^T\lambda$, so $P(g + HPp + A^T\lambda) = Pg + PHPp + 0 = 0$ (since $PA^T = 0$), then p satisfies the optimality conditions of (47). Reversely, if z satisfies $Pg + PHPz = 0$, then $P(g + Hp) = 0$ where $p = Pz$ satisfies then $Ap = 0$, and $P(g + Hp) = (g + Hp) - A^T(AA^T)^{-1}A(g + Hp) = 0$. Defining $\lambda = -(AA^T)^{-1}A(g + Hp)$ leads to the anticipated result.

Then, we use the SCG method to solve (47), where $S = PH^{-1}P$ will be used as scaling matrix. It is interesting to note that $\bar{H}$ and S are are positive definite *in the null space of* A; for any $z \neq 0$ in the null space of A, we have $Pz = z$, so $z^T\bar{H}z = z^TPHPz = z^THz > 0$ because H is positive definite (idem for S). Also $\bar{g} = Pg$ and vectors of the form $\bar{H}z = PHPz$ are in the null space of A. We then call this method PSCG for projected SCG. The following algorithm is an adaptation of the SCG algorithm described in [1] for solving problem (47):

- Start with PSCG iteration index $l = 0$, $z^0 = 0$, residual $r^0 = \bar{g} + \bar{H}z^0 = \bar{g}$, and SCG direction $d^0 = -Sr^0$ (note that z^0, r^0 and d^0 are in the null space of A).
- For iterations $l = 0, 1, ..,$ let

$$\gamma^l = r^{l^T} S r^l / d^{l^T} \bar{H} d^l$$
$$z^{l+1} = z^l + \gamma^l d^l$$
$$r^{l+1} = \bar{g} + \bar{H} z^{l+1}$$
$$\beta^l = r^{l^T} S r^l / r^{l+1^T} S r^{l+1}$$
$$d^{l+1} = -Sr^l + \beta^l d^l$$

By recursivity, z^l, r^l and d^l are clearly all in the null space of A. Therefore, these relations may be rewritten:

- For iterations $l = 0, 1, .., $ let

$$\gamma^l = {r^l}^T H^{-1} r^l / {d^l}^T H d^l$$

$$z^{l+1} = z^l + \gamma^l d^l$$

$$r^{l+1} = \bar{g} + PHz^{l+1}$$

$$\beta^l = {r^l}^T H^{-1} r^l / {r^{l+1}}^T H^{-1} r^{l+1}$$

$$d^{l+1} = -PH^{-1} r^l + \beta^l d^l$$

This approach requires then two projections at the initialization step since $d^0 = -Sr^0 = -PH^{-1}Pr^0 = -PH^{-1}r^0 = -PH^{-1}Pg$, and two projections at each iteration l (in the computation of r and d). Projections may be performed efficiently by the developpements of section 2, and direction $p = z^{l+1}$ is feasible at any iteration index l. Now it remains to show that p is a descent direction if the current solution is not optimal:

Note that if $\bar{g} = Pg = 0$, then $z^0 = 0$ satisfies $Pg + PHPz^0 = 0$, so it is the optimal solution of (47) and therefore of (41) which means that the current solution is the optimal solution of the main problem.

Now we assume that $\bar{g} = Pg \neq 0$. Let l be an iteration index in the PSCG process. If it happens that the residual $r^{l+1} = 0$, then $r^{l+1} = \bar{g} + \bar{H}z^{l+1} = 0$. Therefore, z^{l+1} is the optimal solution of the quadratic problem, and $p = z^{l+1}$ must then satisfy (46). Now, while $r^{l+1} \neq 0$, we have that β^i and γ^i are positive for $i \leq l$. Further, since $p = z^{l+1}$ is in the null space of A, then $g^T p = g^T Pp = \bar{g}^T z^{l+1}$. Under these conditions, let's show that $g^T p = \bar{g}^T z^{l+1} < 0$:

1- For $l = 0$: $g^T p = \bar{g}^T z^{0+1} = \bar{g}^T(z^0 + \gamma^0 d^0) = \bar{g}^T 0 + \gamma^0 \bar{g}^T(-Sr^0) = -\gamma^0 \bar{g}^T S\bar{g}$. But $\gamma^0 > 0$, $Pg = \bar{g} \neq 0$ is in null space of A and $S = PH^{-1}P$ is positive definite in this subspace, then $g^T p = \bar{g}^T z^1 = -\gamma^0 \bar{g}^T S\bar{g} < 0$.

2- For $l \geq 1$: $g^T p = \bar{g}^T z^{l+1} = \bar{g}^T(z^l + \gamma^l d^l) = \bar{g}^T z^l + \gamma^l \bar{g}^T d^l$. Repeating the process with $l, l-1, ..., 1$, leads to $g^T p = \bar{g}^T z^{l+1} = \bar{g}^T z^1 + \sum_{i=1}^{l} \gamma^i \bar{g}^T d^i$.

Since $\bar{g}^T z^1 < 0$, it remains to show that $\sum_{i=1}^{l} \gamma^i \bar{g}^T d^i < 0$.

a- For $i = 1$, we have $\bar{g}^T d^1 = \bar{g}^T(-Sr^0 + \beta^0 d^0) = -\bar{g}^T S\bar{g} - \beta^0 \bar{g}^T S\bar{g} = -\bar{g}^T S\bar{g}(1 + \beta^0)$

b- For $i > 1$, we have $\bar{g}^T d^i = \bar{g}^T(-Sr^{i-1} + \beta^{i-1} d^{i-1}) = -\bar{g}^T Sr^{i-1} + \beta^{i-1} \bar{g}^T d^{i-1}$. But, since $d^0 = -Sr^0 = -S\bar{g}$, and $0 < i - 1$, then $-\bar{g}^T Sr^{i-1} = {d^0}^T r^{i-1} = 0$ since the residual r^{i-1} is orthogonal to all the previous conjugate directions d^j, $j < i - 1$ [1], [11]).

Therefore, for $i > 1$, we have $\bar{g}^\top d^i = \beta^{i-1}\bar{g}^\top d^{i-1} = \beta^{i-1}\beta^{i-2}\bar{g}^\top d^{i-2}$... Then, recursively, $\bar{g}^\top d^i = \prod_{j=1}^{i-1}\beta^j \bar{g}^\top d^1 = -(1+\beta^0)\bar{g}^\top S\bar{g}\prod_{j=1}^{i-1}\beta^j$ (by a-).
Finally, $\sum_{i=1}^{l}\gamma^i\bar{g}^\top d^i = \gamma^1\bar{g}^\top d^1 + \sum_{i>1}^{l}\gamma^i\bar{g}^\top d^i = -\gamma^1(1+\beta^0)\bar{g}^\top S\bar{g} - (1+\beta^0)\bar{g}^\top S\bar{g}\sum_{i>1}^{l}\gamma^i\prod_{j=1}^{i-1}\beta^j$
$= -(\gamma^1 + \sum_{i>1}^{l}\gamma^i\prod_{j=1}^{i-1}\beta^j)(1+\beta^0)\bar{g}^\top S\bar{g} < 0$ since all terms γ, β and $\bar{g}^\top S\bar{g} < 0$ are positive.

3.2 CONSTRAINED MATRIX ESTIMATION AND TRANSPORTATION PROBLEMS

Without bound constraints, the optimal solution of (21) to (23) may be such that some of the variables x_{ij} are negative. In the context of transportation problems (and other practical problems) where the variables represent link flows, negative values of the variables are prohibited while many link flows may be null. The means for describing this situation are nonnegativity constraints (24). The concern of this subsection is to discuss a way to handle nonnegativity constraints (24) (and eventually upper bounds on $x = (x_{ij})$).
In the solution approach that we propose in this paper for this problem, we replace non-negativity constrains (24) by a penalty term (48) that is added to the objective function (21):

$$PT(\mu, x) = \mu\sum_{i=1}^{n}\sum_{j=1}^{m}(\max(0, -x_{ij}))^\nu \tag{48}$$

In penalty function (48), where $\mu \geq 0$ and $\nu \geq 2$, a negative flow x_{ij} is penalized while a nonnegative flow x_{ij} is not. Therefore, when $\mu \to +\infty$, the optimal solution of (49) s.t. (22) and (23), tends to the one of (14) to (17).

$$\min \sum_{i=1}^{n}\sum_{j=1}^{m} f_{ij}(x) + \mu\sum_{i=1}^{n}\sum_{j=1}^{m}(\max(0, -x_{ij}))^\nu = \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}(x) \tag{49}$$

This penalty method is well known and well discribed in most of nonlinear programming references (see for instance [11]). According to the classical method, problem (49) s.t. (22) and (23), must be solved iteratively with a growing sequence $\{\mu_k\}_k$ such that $\lim_{k\to\infty}\mu_k = \infty$. In practice, the process will terminate within some tolerance.

4. NUMERICAL RESULTS

In this section, we present numerical comparative tests obtained by Fortran implementations of the projected gradient algorithm, Newton method (with LDL decomposition and scaled

conjugate gradient method for solving systems of linear equations) and the approximative Newton method based on the projected scaled conjugate gradient technique described in this paper. Codes runned on a SUN 3/50, MC68020/15 Mhz.

Tests was done with bipartite networks (matrices) generated randomly according to the following procedure ([9]): numbers n of origins (rows) and m of destinations (columns) are provided by a user, and then, a matrix $(\bar{x}_{ij})$ is randomly generated in a user supplied interval $[minx, miny]$. Supply S_i, $i = 1, .., n$, and demand D_j, $j = n+1, .., n+m$ are set to $S_i = \sum_j \bar{x}_{ij}$ and $D_j = \sum_i \bar{x}_{ij}$ (this assures that $\sum_i S_i = \sum_j D_j$. Then, the "past observed flows (matrix)" are set to $\bar{x}_{ij} \leftarrow (1 \pm r_{ij})\bar{x}_{ij}$, where r_{ij} is a perturbation factor generated randomly in the interval $[-r, +r]$, r being supplied by a user.

We used these networks (matrices) to test two types of problems:

$$-P1- \qquad \min \sum_i \sum_j (x_{ij} - \bar{x}_{ij})^2 \quad s.t.\ Ax = b$$

$$-P2- \qquad \min \sum_i \sum_j (x_{ij} - \bar{x}_{ij})^4 \quad s.t.\ Ax = b,\ x \geq 0$$

For test problems $P1$, we took $[minx, miny]$=[10,50], and for $P2$ $[minx, miny]$=[0,50]. For all problems $r = 10\%$.

Our stopping criterion is that the projected gradient norm be smaller than 10^{-3}. But, for test problems $P1$, optimality is obtained at the initialization step (see section 3.1); hence, Table 1 presents the CPU time (seconds) required to perform one projection (initialization step) by using the method described in section 3.1-step 0 (column PROJ.3.1), against an LDL decomposition technique. Table 1 demonstrates superiority of the first approach against the latter.

Prob.	Nb. Orig.	Nb. Dest.	Nb. Links	PROJ.3.1 cpu	LDL cpu
1	10	10	100	0.02	0.08
2	50	10	500	0.08	0.14
3	100	10	1000	0.16	0.24
4	50	50	2500	0.38	6.02
5	100	50	5000	0.82	6.42
6	200	50	10000	1.42	7.34
7	100	100	10000	1.42	46.26
8	150	100	15000	2.22	47.76
9	200	100	20000	2.88	49.16

Table 1: Analytic projection vs LDL decomposition

Table 2 gives numerical results obtained with problems of type $P2$, when using first and second order methods described insection 3. For the "NEWTON" columns, LDL and SCG refer to the method used for solving systems of linear equations. As for the Projected SCG method, results appear in column PSCG, where we used $p = z^1$ (see section 3); the method worked better with $p = z^1$ than with a $p = z^l$, $l > 1$. Table 2 shows that PSCG method performed better than the other methods on all the problems, while the projected gradient (PRO.GRA column) and the Newton-SCG methods gave better results than the Newton-LDL decomposition scheme.

Prob.	Nb. Orig.	Nb. Dest.	Nb. Links	PRO.GRA	NEWTON LDL	NEWTON SCG	Quasi-NEWTON PSCG
1	10	10	100	1.84	1.74	1.80	0.84
2	50	10	500	7.00	7.76	7.66	2.16
3	100	10	1000	13.6	17.3	14.9	9.78
4	50	50	2500	24.1	152.	35.7	9.78
5	100	50	5000	67.0	297.	71.2	27.5
6	200	50	10000	126.	716.	119.	38.4
7	100	100	10000	86.0	1251	171.	47.0
8	150	100	15000	165.	1643	261.	59.1
9	200	100	20000	290.	2680	336.	95.3

Table 2: Numerical comparisons for nonlinear constrained problems $P2$.

5. CONCLUSION

In this paper, we consider a bipartite network $G = (O + D, L)$ where O is a set of origin nodes $i = 1,..,n$, D is a set of destination nodes $j = n+1,..,n+m$, and L the set of links (i,j), $i = 1,..,n$, $j = n+1,..,n+m$. The projection of a vector $(u(i,j))$ onto the space defined by the flow conservation constraints at nodes, requires the solution of a quadratic problem, or, equivalently, the solution of a system of linear equations. In this paper, we show that this projection may be done with a linear complexity $O(|L|)$ which is much faster than what was proposed in literature for this problem. This technique is used in combination with the scaled conjugate gradient method to solve variants of the projection problem, where the Hessian matrix is not identity; using the efficient projector described in this paper, we develop a variant of the scaled conjugate gradient method, where all conjugate directions lie in the feasible space defined by the flow conservation constraints at nodes. Theorical results developed in this paper have a wide field of applications, as in nonlinear convex transportation problems, spatial price equilibrium, point to point matrix estimation in telecommunication, etc. Numerical results show that first and second order

(Newton or quasi-Newton) algorithms that use the theorical results developed in this paper, may be much more efficient than other algorithms based on classical conjugate gradient and LDL decomposition techniques.

REFERENCES

[1] Coleman, T.F. (1984) "Large Sparse Numerical Optimization", Lecture Notes in Computer Science 165, Springer-Verlag.

[2] Cottle, R.W., Duvall, S.G. and Zikan, K. (1986) "A Lagrangean relaxation algorithm for the constrained matrix problem", Naval Research Logistics Quarterly 33, 55-76.

[3] Dembo, R.S. and Klinckewicz, J.G. (1981) "A scaled reduced gradient algorithm for network flow problems with convex separable costs", Mathematical Programming Study 15, 125-147.

[4] Drissi-Kaïtouni, O. and Florian M. (1990) " An algorithm for the spatial price equilibrium problem on a general network in the space of the path flows", Journal of Regional Science 31, 1-20.

[5] Florian, M. and Los, M. (1982) "A new look at static price equilibrium models", Regional Science and Urban Economics 12, 579-597.

[6] Florian, M. (1986) "Nonlinear cost network models in transportation analysis" Mathematical Programming Study 26, 167-196.

[7] Guélat, J. and Marcotte, P. (1986) "Some comments on Wolfe's away step", Mathematical Programming 14, 98-107.

[8] Klinckewicz, J.G. (1983) "A Newton method for convex separable network flow problems", Networks 13, 427-442.

[9] Klinckewicz, J.G. (1989) "Implementing an 'exact' Newton methods for separable convex transportation problems", Networks 19, 95-105.

[10] LeBlanc, L.J. (1976) "The conjugate gradient technique for certain quadratic network problems", Naval Research Logistics Quaterly 23, 597-602.

[11] Luenberger, D.G. (1984) Introduction to Linear and Nonlinear Programming, Addison Wesley Publishing Co.

[12] Minoux, M. (1984) "A polynomial algorithm for minimum quadratic cost flow problems", European Journal of Operational Research 18, 377-387.

[13] Nagurney, A. (1987) "Computational comparisons of spatial price equilibrium methods", Journal of Regional Science 27, 55-75.

[14] Nagurney, A. (1990) "Serial and parallel equilibration of large scale constrained matrix problems with application to the social and economic sciences", The International Journal of Supercomputer Applications 4.1.

[15] Rao, R.C. and Shaftel, T.L. (1980) "Computational experience on an algorithm for the transportation problem with nonlinear objective function", Naval Research Logistics Quaterly 27, 145-157.

[16] Takayama, T. and Judge, G.G. (1970) "Alternative spatial price equilibrium models", Journal of Regional Science 10, 1-12.

APPENDIX

We consider here a bipartite network $G = (N, L)$, where N is the set of nodes and L the set of links of G. The set of nodes is partitionned into two disjoint subsets O and D, where O is the set of origin nodes of cardinality $|O| = n$, and D is the set of destination nodes of cardinality $|D| = m$. The set of links L consists of all the pairs $a = (i, j)$, $i = 1, .., n$, $j = n+1, .., n+m$; its cardinality is mn. In the node-link incidence matrix $\bar{A}$ for the bipartite network G each column corresponding to link (i, j), has exactly two nonzero entries that are $+1$ at position j, and -1 at position i. The *truncated* node-link incidence matrix A that we consider is the matrix $\bar{A}$ from which the last row is deleted. We are interested here in solving the system:

$$AA^T\lambda = v \tag{50}$$

In order to solve this system, we will solve first all the following systems:

$$AA^{\top}\lambda^i = e^i \qquad i = 1,..,n+m-1 \tag{51}$$

where e^i is the unit vector having all its entries equal to zero but its i^{th} entry is one.

THEOREM A-Part(1,..,n): *The solution* λ^i *of system* (50) $AA^{\top}\lambda^i = e^i$, $i = 1,..n$, *where* A *is the truncated node-link incidence matrix of a bipartite network* $G=(N,L)$ *defined above is:*

$$\lambda^i_j = \begin{cases} \frac{1}{n} - \frac{1}{nm} + \frac{1}{m} & \text{if } j = 1,..,n, j = i \\ \frac{1}{n} - \frac{1}{nm} & \text{if } j = 1,..,n, j \neq i \\ \frac{1}{n} & \text{if } j = n+1,..,n+m-1 \end{cases} \tag{52}$$

PROOF: We have three types of equations in system (50), that are:

$$\begin{aligned} m\lambda^i_k - \sum_{j=n+1}^{n+m-1} \lambda^i_j &= 0 \qquad k = 1,..,n,\ k \neq i \\ m\lambda^i_i - \sum_{j=n+1}^{n+m-1} \lambda^i_j &= 1 \\ -\sum_{j=1}^{n} \lambda^i_j + n\lambda^i_k &= 0 \qquad k = n+1,..,n+m-1 \end{aligned}$$

From these equations, it appears that all $\lambda^i_j = \overline{\lambda}^i$, $j = 1,..,n$, $,j \neq i$, are equal, and that all $\lambda^i_j = \Delta^i$, $j = n+1,..,n+m-1$, are equal. So these equations may be rewritten:

$$\begin{aligned} m\overline{\lambda}^i - (m-1)\Delta^i &= 0 \qquad k = 1,..,n,\ k \neq i \\ m\lambda^i_i - (m-1)\Delta^i &= 1 \\ -(n-1)\overline{\lambda}^i - \lambda^i_i + n\Delta^i &= 0 \qquad k = n+1,..,n+m-1 \end{aligned}$$

This is a system of three equations with three unknowns, and its solution is:

$$\overline{\lambda}^i = \frac{1}{n} - \frac{1}{nm}$$
$$\lambda^i_i = \frac{1}{n} - \frac{1}{nm} + \frac{1}{m}$$
$$\Delta^i = \frac{1}{n}$$

THEOREM A-Part$(n+1,..,n+m-1)$: *The solution* λ^i *of system* (50) $AA^\top\lambda^i = e^i$, $i = n+1,..n+m-1$, *where A is the truncated node-link incidence matrix of a bipartite network G=(N,L) defined above is:*

$$\lambda^i_j = \begin{cases} \frac{1}{n} & \text{if } j = 1,..,n+m-1, j \neq i \\ \frac{2}{n} & \text{if } j = 1,..,n+m-1, j = i \end{cases} \tag{53}$$

PROOF: In this case, we also have three types of equations in system (50), that are:

$$\begin{aligned} m\lambda^i_k - \sum_{j=n+1}^{n+m-1} \lambda^i_j &= 0 \quad k = 1,..,n \\ -\sum_{j=1}^{n} \lambda^i_j + n\lambda^i_k &= 0 \quad k = n+1,..,n+m-1,\ k \neq i \\ -\sum_{j=1}^{n} \lambda^i_j + n\lambda^i_i &= 1 \end{aligned}$$

From these equations, it appears that all $\lambda^i_j = \overline{\lambda}^i$, $j = 1,..,n$, are equal, and that all $\lambda^i_j = \underline{\lambda}^i$, $j = n+1,..,n+m-1$, $j \neq i$, are equal. So these equations may be rewritten:

$$\begin{aligned} m\overline{\lambda}^i - (m-2)\underline{\lambda}^i - \lambda^i_i &= 0 \quad k = 1,..,n \\ -n\overline{\lambda}^i + n\underline{\lambda}^i &= 0 \quad k = n+1,..,n+m-1, k \neq i \\ -n\overline{\lambda}^i + n\lambda^i_i &= 1 \end{aligned}$$

This is a system of three equations with three unknowns, and its solution is:

$\overline{\lambda}^i = \underline{\lambda}^i = \frac{1}{n}$

$\lambda^i_i = \frac{2}{n}$

•

We must note here that since $\lambda^i = (AA^\top)^{-1}e^i$, $i = 1,..,n+m-1$, then λ^i, $i = 1,..,n+m-1$ are the columns of $(AA^\top)^{-1}$. Therefore, this matrix may be used explicitly to solve any system of the form $\lambda = (AA^\top)^{-1}v$. Since $v = \sum_{i=1}^{n+m-1} v_i e^i$ where v_i is the i^{th} entry of v, then, $AA^\top\lambda = v = \sum_{i=1}^{n+m-1} v_i e^i$, and $\lambda = \sum_{i=1}^{n+m-1} v_i\lambda^i$. By Theorem A, λ^i, $i = 1,..,n+m-1$ are known, so it is easy to do the summation over $i = 1,..,n+m-1$ of $v_i\lambda^i$, which leads exactly to the vector λ of Theorem 1 in section 2.

$$(AA^{\top})^{-1} = \begin{array}{|c|c|}\hline Xii & Xij \\ \hline Xji & Xjj \\ \hline \end{array}$$

where:

$$Xii = (1/n - 1/nm)U + (1/m)I_n$$

$$Xjj = (1/n)(U + I_{m-1})$$

$$Xij = (1/n)U$$

$$Xji = Xij^{\top}$$

where I_n and I_{m-1} are identity matrices of size $n \times n$ and $(m-1) \times (m-1)$, and U is unity matrix (all elements are 1) of size $n \times (m-1)$.

Note that one can check that $AA^{\top}\lambda$, where λ is given by (18), is in effect equal to v:

• For indices $k = 1, .., n$:

$$\begin{aligned}
\sum_{j=1}^{n+m-1}(AA^{\top})_{kj}\lambda_j &= \sum_{j=1}^{n}(AA^{\top})_{kj}\lambda_j + \sum_{j=n+1}^{n+m-1}(AA^{\top})_{kj}\lambda_j \\
&= m\lambda_k + \sum_{j=n+1}^{n+m-1}(-1)\lambda_j \\
&= m\frac{1}{n}(\bar{v} - \frac{1}{m}\bar{v} + \frac{n}{m}v_k) + (-1)\sum_{j=n+1}^{n+m-1}\frac{1}{n}(\bar{v} + v_j) \\
&= \frac{m}{n}\bar{v} - \frac{1}{n}\bar{v} + v_k - \sum_{j=n+1}^{n+m-1}\frac{1}{n}\bar{v} - \frac{1}{n}\sum_{j=n+1}^{n+m-1}v_j \\
&= \frac{m}{n}\bar{v} - \frac{1}{n}\bar{v} + v_k - (m-1)\frac{1}{n}\bar{v} - \frac{1}{n}(\bar{v} - \bar{v}) \\
&= \frac{m}{n}\bar{v} - \frac{1}{n}\bar{v} + v_k - \frac{m}{n}\bar{v} + \frac{1}{n}\bar{v} - \frac{1}{n}\bar{v} + \frac{1}{n}\bar{v} \\
&= v_k
\end{aligned}$$

• For indices $k = n+1, .., n+m-1$:

$$\begin{aligned}
\sum_{j=1}^{n+m-1}(AA^{\top})_{kj}\lambda_j &= \sum_{j=1}^{n}(AA^{\top})_{kj}\lambda_j + \sum_{j=n+1}^{n+m-1}(AA^{\top})_{kj}\lambda_j \\
&= \sum_{j=1}^{n}(-1)\lambda_j + n\lambda_k \\
&= -\sum_{j=1}^{n}\frac{1}{n}(\bar{v} - \frac{1}{m}\bar{v} + \frac{n}{m}v_j) + n\frac{1}{n}(\bar{v} + v_k)
\end{aligned}$$

$$
\begin{aligned}
&= -n\frac{1}{n}(\bar{v} - \frac{1}{m}\bar{v}) - \frac{1}{m}\sum_{j=1}^{n} v_j + n\frac{1}{n}(\bar{v} + v_k) \\
&= -(\bar{v} - \frac{1}{m}\bar{v}) - \frac{1}{m}\bar{v} + \bar{v} + v_k \\
&= v_k
\end{aligned}
$$